REMARKS

In the above-identified Office Action, the Examiner rejected Claims 1 - 4, 8 - 11 and 15 - 18 under 35 U.S.C. §103(a) as being unpatentable over Bonwick in view of Printezis et al. Claims 6, 7, 13, 14, 20 and 21 were objected to for being dependent upon a rejected claim but would be allowable if rewritten in independent form to include all the limitations of the base claim and any intervening claim.

The Examiner is thanked for the telephone interview of June 12, 2006. In that interview Claim 1 and the applied references were discussed. However, no agreements were reached.

By this amendment, Claims 1 – 21 remain pending in the Application. For the reasons stated more fully below, Applicants submit that the pending claims are allowable over the applied references. Hence, reconsideration, allowance and passage to issue are respectfully requested.

As stated in the SPECIFICATION, presently, memory fragmentation is one of the biggest problems facing modern memory management subsystems. Memory fragmentation occurs when system memory has a multiplicity of partially-used allocated blocks of space.

One of the methods that has been used to reduce memory fragmentation is to allocate memory space in slabs. A slab is an allocated space (e.g., a block of 16 contiguous pages) of memory that is dedicated to hold only fixed sized objects (i.e., one type of data). For example, slabs may be allocated to hold only inodes. An inode is a data structure that contains system information about a file with which it is associated. When inodes are held in a slab of memory, they are in one area of the memory instead of being scattered all around. This helps in reducing memory fragmentation.

When the system is in need of a large section of memory space and none is available, the system may decide to de-allocate a slab. However, since a block of allocated memory space may not be de-allocated until empty, a need

PAGE 10/13 * RCVD AT 6/15/2006 4:22:10 PM [Eastern Daylight Time] * SVR:USPTO-EFXRF-2/6 * DNIS:2738300 * CSID:512 306 0240 * DURATION (mm-ss):03-36

therefore exists for a method of squeezing slabs of memory empty in order to facilitate de-allocation.

The present invention provides such a method. In accordance with the teachings of the invention, when a slab of memory is to be squeezed empty, data will be precluded from being placed in any empty locations that it may have.

The invention is set forth in claims of varying scopes of which Claim 1 is illustrative.

> 1. A method of squeezing slabs empty, a slab being a block of allocated memory space, the method comprising the steps of:
> ***determining whether a slab is to be squeezed empty***; and
> ***precluding, if the slab is to be squeezed empty, data from being placed in any unused space of the slab***. (Emphasis added.)

The Examiner rejected the independent claims under 35 U.S.C. §103(a) as being unpatentable over Bonwick in view of Printezis et al. Applicants respectfully disagree.

Bonwick discloses a slab memory allocator. According to the teachings of Bonwick, allocation and de-allocation of objects are among the most common operations in the kernel. Thus, a fast kernel memory allocator becomes essential to enhance performance. Consequently, Bonwick devises a method of allocating memory space in slabs (a slab is, for example, a block of 16 contiguous pages) instead of allocating one page of memory at a time. As in the case of pages, when a slab becomes empty, it is or can be deallocated.

However, Bonwich does not teach, show or so much as suggest ***a method of squeezing*** (i.e., forcing) ***a slab empty***. As Bonwick stated (see lines 13 – 15 on paragraph 3.2; see also page 2, lines 5 and 6 of the Specification), a slab of memory may not be de-allocated until it is empty. Thus, the present Invention squeezes or pushes the slab to become empty such that it

PAGE 11/13 * RCVD AT 6/15/2006 4:22:10 PM [Eastern Daylight Time] * SVR:USPTO-EFXRF-2/6 * DNIS:2738300 * CSID:512 306 0240 * DURATION (mm-ss):03-36

can be de-allocated sooner than it would otherwise. By contrast, Bonwick just waits for the slab to become empty.

Since Bonwick does not teach a method of squeezing a slab empty, he cannot have taught the step of ***determining whether a slab is to be squeezed empty*** as claimed.

Further, Bonwick does teach, show or suggest the step of ***precluding, if the slab is to be squeezed empty, data from being placed in any unused space of the slab***.

Regarding the Printezis et al. reference, it teaches a mostly concurrent compaction in a garbage collection system. Specifically, memory that has been allocated to a program is reclaimed by first identifying variables containing pointers that point to objects stored in a selected subset of memory. Identification of the relevant pointers, as well as addition of their identities to a data structure, may be performed concurrently with execution of the program. Concurrently with these steps, a write barrier marks regions of memory in which one or more pointers have been modified by the program. The write barrier operates, for example, by setting "dirty" bits in a "card table" having entries corresponding to a number of predefined memory regions.

Also while the program is executing, a number of locations outside the subset of memory are identified which will later be used to store the objects located in the subset of memory.

Execution of the program is then suspended. The memory regions marked as "dirty" are examined to identify any further variables pointing to objects located in the subset of memory. Indications of any such pointers are added to the data structure. Those variables indicated by the data structure are then examined to determine whether they continue to point to objects within the subset of memory. Those that continue to do so are modified to point to corresponding locations outside of the subset of memory. The objects are then copied to the locations outside of the subset of memory, and the program is restarted. The subset of memory in which the relocated objects were stored can

PAGE 12/13 * RCVD AT 6/15/2006 4:22:10 PM [Eastern Daylight Time] * SVR:USPTO-EFXRF-2/6 * DNIS:2738300 * CSID:512 306 0240 * DURATION (mm-ss):03-36

now be added to a free list for reallocation. The subset of memory can be selected such that a large portion of contiguous memory space results when the memory objects within it are deallocated. In this way a significant amount of compaction can be accomplished, thus enabling the system to subsequently satisfy large memory allocation requests.

Thus, Printezis et al. teach a method of de-allocating or re-claiming a region in memory by moving data in that region to another region of the memory. While doing so, Printezis et al. ensure that modified pointers do not point to data in that region of memory. But Printezis et al. do not teach the steps of ***determining whether a slab is to be squeezed empty***; and ***precluding, if the slab is to be squeezed empty, data from being placed in any unused space of the slab***.

Since neither Bonwick nor Printezis et al. teach the amboledened/italicized limitations in the above-reproduced Claims 1, Applicants submit that Claim 1, along with its dependent claims, is allowable over the references. The other independent claims (i.e., Claims 8, and 15), which all include the above-emboldened/italicized limitations, as well as their dependent claims, are also allowable over the references. Consequently, Applicants once more respectfully request reconsideration, allowance and passage to issue of the claims in the application.

Respectfully Submitted

By: _____
Volel Emile
Attorney for Applicants
Registration No. 39,969
(512) 306-7969

PAGE 13/13 * RCVD AT 6/15/2006 4:22:10 PM [Eastern Daylight Time] * SVR:USPTO-EFXRF-2/6 * DNIS:2738300 * CSID:512 306 0240 * DURATION (mm-ss):03-36